About the Cloud SQL Proxy

MySQL (https://cloud.google.com/sql/docs/mysql/sql-proxy) | PostgreSQL (https://cloud.google.com/sql/docs/postgres/sql-proxy) | **SQL Server**

Beta

This feature is in a pre-release state and might change or have limited support. For more information, see the product launch stages (https://cloud.google.com/products/#product-launch-stages).

This page provides a basic introduction to the Cloud SQL Proxy, and describes the proxy options.

What the proxy provides

The **Cloud SQL Proxy** provides secure access to your Cloud SQL Second Generation instances without having to whitelist IP addresses (https://cloud.google.com/sql/docs/sqlserver/configure-ip) or configure SSL (https://cloud.google.com/sql/docs/sqlserver/configure-ssl-instance).

Accessing your Cloud SQL instance using the Cloud SQL Proxy offers these advantages:

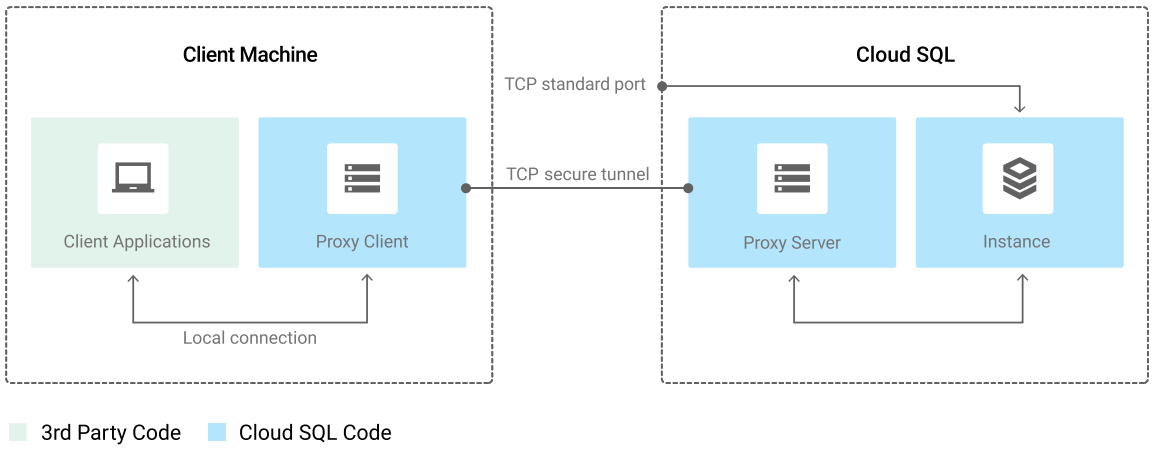
- **Secure connections:** The proxy automatically encrypts traffic to and from the database using TLS 1.2 with a 128-bit AES cipher; SSL certificates are used to verify client and server identities.
- **Easier connection management:** The proxy handles authentication with Cloud SQL, removing the need to provide static IP addresses.

The proxy does not provide a new connectivity path; it relies on existing IP connectivity. For example, you cannot use the proxy to connect with an instance using Private IP (https://cloud.google.com/sql/docs/sqlserver/private-ip) unless the proxy is using a VPC network that has been configured for private services access.

How the Cloud SQL Proxy works

The Cloud SQL Proxy works by having a local client, called the proxy, running in the local environment. Your application communicates with the proxy with the standard database protocol used by your database. The proxy uses a secure tunnel to communicate with its companion process running on the server.

The following diagram shows how the proxy connects to Cloud SQL:



Requirements for using the Cloud SQL Proxy

To use the proxy, you must meet the following requirements:

- The Cloud SQL Admin API must be enabled.
- You must provide the proxy with Google Cloud authentication credentials (#authentication-options).
- You must provide the proxy with a valid database user account and password.
- The instance must either have a public IPv4 address, or be configured to use private IP (https://cloud.google.com/sql/docs/sqlserver/private-ip).

  The public IP address does not need to be accessible to any external address (whitelisted).

Installing the Cloud SQL Proxy

LINUX 64-BIT | LINUX 32-BIT | MORE

1. Download the proxy:

```
wget https://dl.google.com/cloudsql/cloud_sql_proxy.linux.amd64 -O cloud_sql_
```

2. Make the proxy executable:

```
chmod +x cloud_sql_proxy
```

If your operating system isn't included here, you can also compile the proxy from source (http://github.com/GoogleCloudPlatform/cloudsql-proxy).

Proxy startup options

When you start the proxy, you provide it with the following information:

- What Cloud SQL instances it should establish connections to
- Where it will listen for data coming from your application to be sent to Cloud SQL
- Where it will find the credentials it will use to authenticate your application to Cloud SQL
- If required, which IP address type to use.

The proxy startup options you provide determine whether it will listen on a TCP port or on a Unix socket. If it is listening on a Unix socket, it creates the socket at the location you choose; usually, the /cloudsql/ directory. For TCP, the proxy listens on `localhost` by default.

Note: The Cloud SQL Proxy does not support Unix sockets on Windows.

You can install the proxy anywhere in your local environment. The location of the proxy binaries does not impact where it listens for data from your application.

Using a service account for authentication

The proxy requires authentication. The advantage of using a service account for this purpose is that you can create a credential file specifically for the proxy, and it is explicitly and permanently linked to the proxy as long as it is running. For this reason, this is the recommended method for production instances not running on a Compute Engine instance.

The credential file can be duplicated in a system image if you need to invoke the Cloud SQL Proxy from multiple machines.

To use this method, you must create and manage the credential file. Only users with the `resourcemanager.projects.setIamPolicy` permission (such as project owners) can create the service account. If your Google Cloud user does not have this permission, you must have someone else create the service account for you, or use another method to authenticate the proxy.

For help with creating a credential file, see Creating a service account (#create-service-account).

Options for authenticating the Cloud SQL Proxy

Note: When you authenticate the Cloud SQL Proxy, you enable it to access Google Cloud on behalf of your application, using a set of Google credentials. This is separate from database user authentication.

The Cloud SQL Proxy provides several alternatives for authentication, depending on your environment. The proxy checks for each of the following items, in this order, using the first one it finds to attempt to authenticate:

1. **Credentials supplied by the credential_file flag.**

   Use a service account (#create-service-account) to create and download the associated JSON file, and set the `-credential_file` flag to the path of the file when you start the Cloud SQL Proxy. The service account must have the required permissions (#permissions) for the Cloud SQL instance.

2. **Credentials supplied by an access token.**

Create an access token (http://cloud.google.com/storage/docs/json_api/v1/how-tos/authorizing#AboutAuthorization) and provide it on the command line with the `-token` flag when you start the Cloud SQL Proxy .

3. **Credentials supplied by an environment variable.**

   You can set the `GOOGLE_APPLICATION_CREDENTIALS` environment variable to the path of the json key file created for a service account (#create-service-account).

4. **Credentials from an authenticated Cloud SDK client.**

   If you have installed the Cloud SDK and used it to authenticate to Google Cloud, the Cloud SQL Proxy can use the same credentials. This method is especially helpful for getting a development environment up and running quickly.For a production environment, you should use one of the other methods to authenticate.

   ★ To enable the proxy to use your Cloud SDK credentials, you must use the **`gcloud auth login`** command to authenticate the Cloud SDK.

   You can determine what your current Cloud SDK credentials are by using the **`gcloud auth list`** command.

5. **Credentials associated with the Compute Engine instance.**

   If you are connecting to Cloud SQL from a Compute Engine instance, the proxy can use the service account associated with the Compute Engine instance. If the service account has the required permissions (#permissions) for the Cloud SQL instance, the proxy authenticates successfully.

   If the Compute Engine instance is in the same project as the Cloud SQL instance, the default service account for the Compute Engine instance has the necessary permissions for authenticating the proxy. If the two instances are in different projects, you must add the Compute Engine instance's service account to the project containing the Cloud SQL instance.

Required permissions for service accounts

When you use a service account to provide the credentials for the proxy, you must create it with sufficient permissions. If you are using the finer-grained Identity Access and Management (https://cloud.google.com/sql/docs/sqlserver/project-access-control) (IAM) roles to manage your Cloud SQL permissions, you must give the service account a role that includes the `cloudsql.instances.connect` permission. The predefined Cloud SQL roles that include this permission are:

- Cloud SQL Client
- Cloud SQL Editor
- Cloud SQL Admin

If you are using the legacy project roles (Viewer, Editor, Owner), the service account must have at least the Editor role.

Options for specifying Cloud SQL instances

There are several ways to tell the proxy which instances you want to connect to. Some are explicit and some are implicit. In some configurations, you do not have to tell the proxy ahead of time which instances you want to connect to, because the proxy connects based on connection requests.

Your options for instance specification depend on your operating system and environment:

Option	Benefits	Caveats and Requirements	Linux/macOS (Unix sockets)	Java	Windows	Notes
Automatic instance discovery	No need to specify instances; sockets created for all instances in default project.	Proxy API usage is increased. Must have Cloud SDK installed and authenticated, with a default project set. Must restart proxy to add new instance.	Supported	No	No	Not recommended for production
Project	No need	Proxy API	Supported	No	No	Use `-projects` parameter. Not re

discovery	to specify instances; sockets created for all instances in specified projects.	usage is increased. Must have Cloud SDK installed and authenticated. Must restart proxy to add new instance.				production instances.
Instances specified on proxy invocation	Instance list known and static.	Must restart proxy to add new instance.	Supported	Supported with TCP sockets	Supported with TCP sockets	Use **-instances** parameter. For m instances, use a comma-separated spaces. Learn more (#multiple-ins
Instances specified using Compute Engine metadata	Instance list can be updated by changing the metadata value without restarting the proxy.	Available only on Compute Engine.	Supported	Supported with TCP sockets	Supported with TCP sockets	Use **-instances_metadata** flag. (https://cloud.google.com/compu retrieving-metadata#default) .

See sample invocations and connection strings (#invocations).

Using the proxy with private IP

The proxy establishes a connection with your Cloud SQL instance using IP. By default, the proxy attempts to connect using a public IPv4 address. If the proxy is using the same VPC network as the Cloud SQL instance, and the instance has a private IP address, the proxy can connect using private IP.

If your Cloud SQL instance has only private IP, the proxy uses the private IP address to connect. If the instance has both public and private IP configured, and you want the proxy to use the private IP address, you must provide the following option when you start the proxy:

```
-ip_address_types=PRIVATE
```

Tips for working with Cloud SQL Proxy

Invoking the Cloud SQL Proxy

All of the example proxy invocations start the proxy in the background, so a prompt is returned. It is preferable to reserve that terminal for the proxy, to avoid having its output mixed with the output from other programs. Also, the output from the proxy can help you diagnose connection problems, so it can be helpful to capture in a log file. If you do not start the proxy in the background, the output goes to stdout unless redirected.

You do not have to use `/cloudsql` as the directory for the proxy sockets. (That directory name was chosen to minimize differences with App Engine connection strings.) If you change the directory name, however, keep the overall length to a minimum; it is incorporated in a longer string that has a length limit imposed by the operating system.

Using the proxy to connect to multiple instances

You can use one local proxy client to connect to multiple Cloud SQL instances. The way you do this depends on whether you are using Unix sockets or TCP.

Unix sockets

To connect the proxy to multiple instances, you provide the instance connection names with the `-instances` parameter, in a comma-separated list (no spaces). The proxy connects to each instance when it starts.

You connect to each instance using its socket, in the specified directory.

TCP

When you connect using TCP, you specify the port on your machine to use to connect to the instance, and every instance must have its own port. The sqlcmd tool uses 1433 by default, but you can specify another port for it to use.

For example:

```
./cloud_sql_proxy -instances=myProject:us-central1:myInstance=tcp:1433,myProject:us-
sqlcmd -U myUser -S "127.0.0.1,1434"
```

Keeping the Cloud SQL Proxy up to date

Google occasionally releases new versions of the proxy. You can see what the current version is by checking the Cloud SQL Proxy GitHub releases page (https://github.com/GoogleCloudPlatform/cloudsql-proxy/releases). Future proxy releases will also be noted in the Google Groups Cloud SQL announce (https://groups.google.com/forum/#!forum/google-cloud-sql-announce) forum.

Note: you must be running version 1.12 or later to connect using private IP (https://cloud.google.com/sql/docs/sqlserver/private-ip).

API usage

The Cloud SQL Proxy issues requests to the Cloud SQL API. These requests count against the API quota for your project.

The highest API usage occurs when you start the proxy; this is especially true if you use automatic instance discovery or the `-projects` parameter. While the proxy is running, it issues 2 API calls per hour per connected instance.

Cloud SQL Proxy parameters and flags

The Cloud SQL Proxy accepts several flags and parameters when it is started. These options determine where and how the Cloud SQL Proxy creates the sockets it uses for communicating with Cloud SQL, and how it authenticates.

For help with proxy options, see the following information:

- Options for authenticating the Cloud SQL Proxy (#authentication-options)
- Options for specifying Cloud SQL instances (#instances-options)
- Example proxy invocations (#invocations)
- Cloud SQL Proxy GitHub page (https://github.com/GoogleCloudPlatform/cloudsql-proxy)
- The proxy help, displayed with `./cloud_sql_proxy -help`

Proxy invocations and sqlcmd client connection strings

You can use proxy invocations and connection strings, for example, in commands for a SQL Server user `myUser`, for the `myInstance` instance, located in `us-central1`, in the `myProject` project.

For more information about Cloud SQL Proxy options and connection strings, see the Cloud SQL Proxy GitHub page (https://github.com/GoogleCloudPlatform/cloudsql-proxy).

Creating a service account and generating a key file

To create a service account:

Note: To create a service account with the required permissions, you must have `resourcemanager.projects.setIamPolicy` permission. This permission is included in the Project Owner, Project IAM Admin, and Organization Administrator roles.
You must also have enabled the Cloud SQL Admin API.

1. Go to the **Service accounts** page of the Google Cloud Console.

   GO TO THE SERVICE ACCOUNTS PAGE (HTTPS://CONSOLE.CLOUD.GOOGLE.COM/IAM-ADMIN/SERV
2. Select the project that contains your Cloud SQL instance.
3. Click **Create service account**.
4. In the **Create service account** dialog, provide a descriptive name for the service account.
5. For **Role**, select one of the following roles:
   - **Cloud SQL > Cloud SQL Client**
   - **Cloud SQL > Cloud SQL Editor**
   - **Cloud SQL > Cloud SQL Admin**

   Alternatively, you can use the primitive Editor role by selecting **Project > Editor**, but the Editor role includes permissions across Google Cloud.

   If you do not see these roles, your Google Cloud user might not have the `resourcemanager.projects.setIamPolicy` permission. You can check your permissions by going to the IAM page (https://console.cloud.google.com/iam-admin) in the Google Cloud Console and searching for your user id.
6. Change the **Service account ID** to a unique, easily recognizable value.

7. Click **Furnish a new private key** and confirm that the key type is `JSON`.
8. Click **Create**.

   The private key file is downloaded to your machine. You can move it to another location. Keep the key file secure.

For more information about service accounts, see the Authentication documentation (https://cloud.google.com/docs/authentication#service_accounts).

Using the Cloud SQL Proxy in a production environment

When you are using the Cloud SQL Proxy in a production environment, there are some steps you can take to ensure that the proxy provides the required availability for your application.

Ensure that the Cloud SQL Proxy is run as a persistent service

If the proxy process is terminated, all existing connections through it are dropped, and your application cannot create any more connections to the Cloud SQL instance with the proxy. To prevent this scenario, be sure to run the proxy as a persistent service, so that if the proxy exits for any reason, it is automatically restarted. This can be accomplished by using a service such as `systemd`, `upstart`, or `supervisor`. For the Windows operating system, run the proxy as a Windows Service. In general, the proxy should have the same uptime requirements as your application process.

Note: The Cloud SQL Proxy is a Windows executable but is not a native Windows Service. There are several tools available that can wrap a regular application binary as a service.

How many copies of the Cloud SQL Proxy your application needs

There is no need to create a proxy process for every application process; many application processes can share a single proxy process. In general, you should run one proxy client process per workstation or virtual machine.

If you are using auto-scaling for virtual machines, ensure that the proxy is included in your virtual machine configuration, so that whenever a new virtual machine is started, it has its own proxy process.

It is up to you to manage how many connections your application requires, whether by limiting or pooling the connections. The proxy does not place any limitations on new connection rates or persistent connection count.

Reducing Cloud SQL Proxy output

If you need to reduce the size of the proxy log, you can do so by setting `-verbose=false` when you start the proxy. Keep in mind, however, that doing so will reduce the effectiveness of the proxy output in diagnosing connection issues.

How failover affects the Cloud SQL Proxy

If you are running the proxy on an instance configured for High Availability, and a failover occurs, connections through the proxy are affected the same way as connections over IP: all existing connections are lost, and the application must establish new connections. However, no manual intervention is required; the application can continue using the same connection strings it was before.

Troubleshooting Cloud SQL Proxy connections

If you are having trouble connecting to your Cloud SQL instance using the Cloud SQL Proxy, here are a few things to try to find what's causing the problem.

- Check the proxy output.

  Often, the proxy output can help you determine the source of the problem and how to solve it. Pipe the output to a file, or watch the terminal where you started the proxy.

- If you are getting a `403 notAuthorized` error, and you are using a service account to authenticate the proxy, make sure the service account has the correct permissions (#permissions).

  You can check the service account by searching for its ID on the IAM page (https://console.cloud.google.com/iam-admin). It should have the `cloudsql.instances.connect` permission. All Cloud SQL predefined roles have this permission, except for `Cloud SQL Viewer`. In addition, the legacy project roles of `Editor` and `Owner` have the required permission.

- Make sure the Cloud SQL API is enabled.

If it is not, you will see output like `Error 403: Access Not Configured` in your proxy logs.

- If you are including multiple instances in your instances list, make sure you are using a comma as a delimiter, with no spaces. If you are using TCP, make sure you are specifying different ports for each instance.
- If you are attempting to connect from an application, connect using an administrative client (https://cloud.google.com/sql/docs/sqlserver/connect-admin-proxy) first, to eliminate any issues with your application.
- If you are connecting using UNIX sockets, confirm that the sockets were created by listing the directory you provided when you started the proxy.
- If you have an outbound firewall policy, make sure it allows connections to port 3307 on the target machine.

★ **Note:** Port 3307 is used by the Cloud SQL Proxy to connect to the proxy server.

What's next

- Learn more about the Cloud SQL Proxy (https://github.com/GoogleCloudPlatform/cloudsql-proxy).
- Connect a sqlcmd client using the proxy from a client machine (https://cloud.google.com/sql/docs/sqlserver/connect-admin-proxy).